AMENDMENTS TO THE CLAIMS

Please amend the Claims as follows:

1. (Currently Amended) A method for compiling a structured document schema into type annotation records comprising steps of:

a. building a type hierarchy ordered tree from a structured document schema from type record wherein each of said type records contains typing tuples,

b. creating a typing set containing said typing tuples in said type hierarchy ordered tree,

c. creating an ambiguity typing sequence for said typing tuples sharing a common first field and having a unique second field,

d. arranging said ambiguity typing sequence based on an offset number assigned to a third field of each of said typing tuples in said ambiguity typing sequence,

e. extracting a second field from each of said typing tuples accorded to sorted order of said ambiguity typing sequences, and

f. creating a type indexing data structure populated with said extracted second field to map each type name to a type.

2. (Original) A method for the compilation of a structured document schema, as per claim 1, wherein said structured document schema is an XML document schema

3. (Original) A method for the compilation of a structured document schema, as per claim 1, wherein said typing tuples in said typing set are sorted to create said ambiguity typing sequence.

4. (Original) A method for the compilation of a structured document schema, as per claim 1, wherein said arranging step is further comprised of: collecting each third field of said typing tuples and sorting said typing tuples in said ambiguity sequence with respect to third field of said typing tuple.

5. (Original) A method for the compilation of a structured document, as per claim 1, wherein a typing tuple is comprised of an element type name in said first field, a type identifier in said second field, and a parent element name in said third field.

6. (Original) A method for the compilation of a structured document, as per claim 5, wherein said name in said first field is used in said sorting step to alphabetically sort said typing tuples in typing set.

7. (Original) A method for the compilation of a structured document, as per claim 5, wherein said name is one of: a type name, element name, or attribute name; and said type identifier is one of: a type, element, or attribute.

8. (Original) A method for the compilation of a structured document, as per claim 5, wherein said third field is empty if said parent element name corresponds to a global element type.

9. (Original) A method for the compilation of a structured document, as per claim 5, wherein a typing set is comprised of distinct typing tuples, wherein two typing tuples are distinct if either said first fields of both of said typing tuples are different or said second fields of both of said typing tuples are different.

10. (Original) A method for the compilation of a structured document, as per claim 1, wherein said offset in said arranging is the position of said ambiguity type in an ambiguity typing sequence.

11. (Original) A method for the compilation of a structured document, as per claim 1, wherein said type indexing data structure can be any one of: a hash table, a binary tree, and a B+ tree.

12. (Original) A method for the compilation of a structured document, as per claim 1, wherein said type indexing data structure is comprised of a column indicating ambiguity type for each of said type names and a column indicating offset.

13. (Currently Amended) A method for a database engine to perform type annotation of structured documents or structured document fragments in the absence of full schema validation, comprising steps of:

a. building a type annotation data structure comprising a structured document type hierarchy, a type indexing data structure, and a type array,

b. mapping a type name string to each element type in said structured document type hierarchy, and

c. annotating a structured document or fragment using type annotation records obtained from said type annotation data structure and said type name mapping.

14. (Currently Amended) A method for a database engine to perform type annotation, as per claim 13, wherein said mapping step further comprises steps of:

a. loading said type annotation data structure into a runtime validation engine,

b. creating an empty offset stack data structure,

c. pushing record containing a value of zero onto said offset stack,

d. using a token from an XML document or document fragment to key a search on a type indexing data structure to determine an index for said token,

e. incrementing said index by value in topmost record of offset stack if said token is indicated to be of ambiguous type, and

f. indicating element type in a type array at said index location.

15. (Original) A method for annotating type, as per claim 14, wherein said type is an XML type.

16. (Original) A method for annotating type, as per claim 14, wherein said record is a type annotation record.

17. (Original) A method for annotating type, as per claim 14, wherein said method supports defaults, "any" type, and "xsi:nil='true'" attribute.

18. (Original) A method for annotating type, as per claim 17, wherein attribute defaults are supported by associating attribute types with element types in said type annotation records.

19. (Original) A method for annotating type, as per claim 17, wherein a type is annotated with "any" type if an index is not located for said token in said searching step.

20. (Original) A method for annotating type, as per claim 17, wherein said method is not performed if an "xsi:nil='true' attribute is encountered.

21. (Original) A method for annotating type, as per claim 14, wherein said token comprises any of: a start tag and element name; a start tag, element name, and type name; an attribute type and attribute name; or an end tag.

22. (Original) A method for annotating type, as per claim 14, wherein said ambiguous type of said token is determined by a consultation of said typing array.

23. (Original) A method for annotating type, as per claim 14, wherein a record is pushed onto said offset stack if said token is either a start tag and element name; or a start tag, element name, and type name.

24. (Original) A method for annotating type, as per claim 14, wherein if said token is an end tag; a topmost record of said offset stack is removed.

25. (Currently Amended) An article of manufacture comprising a computer usable medium having computer readable program code embodied therein which implements the compilation of a structured document schema into type annotation records comprising modules to execute the steps of:

a. building a type hierarchy ordered tree from a structured document schema from type record wherein each of said type records contains typing tuples,

b. creating a typing set containing said typing tuples in said type hierarchy ordered tree,

c. creating an ambiguity typing sequence for said typing tuples sharing a common first field and having a unique second field,

d. arranging said ambiguity typing sequence based on an offset number assigned to a third field of each of said typing tuples in said ambiguity typing sequence,

e. extracting a second field from each of said typing tuples accorded to sorted order of said ambiguity typing sequences, and

f. creating a type indexing data structure populated with said extracted second field to map each type name to a type.

26. (Currently Amended) An article of manufacture comprising a computer usable medium having computer readable program code embodied therein which comprising modules to execute the steps of:

a. loading type annotation data structure into a runtime validation engine,

b. creating an empty offset stack data structure,

c. pushing a record containing a value of zero onto said offset stack,

d. using a token from an XML document or document fragment to key a search on a typing index to determine an index for said token,

e. incrementing said index by value in topmost record of offset stack if said token is indicated to be of ambiguous type, and

f. indicating element type in a typing array at said index location.